

# Improving SpellChecking: an effective Ad-Hoc probabilistic lexical measure for general typos

Hicham Gueddah<sup>1</sup>, Mohamed Nejja<sup>2</sup>, Said Iazzi<sup>3</sup>, Abdellah Yousfi<sup>4</sup>, Si Lhoussain Aouragh<sup>2</sup>

<sup>1</sup>Intelligent Processing and Security of Systems team- FSR, E.N.S, Mohammed V University, Rabat, Morocco

<sup>2</sup>E.N.S.I.A.S, Mohammed V University, Rabat, Morocco

<sup>3</sup>Faculty of Sciences, Mohammed V University, Rabat, Morocco

<sup>4</sup>Faculty of Law, Economics and Social Sciences-Souissi, Mohammed V University, Rabat, Morocco

## Article Info

### Article history:

Received Oct 11, 2021

Revised Apr 16, 2022

Accepted May 3, 2022

### Keywords:

Blank-space

Correction

Insertion

Measure

Missing

Probability

Spelling

## ABSTRACT

Since the era of learning to write by human beings, mistakes made in typing words have occupied a privileged place in linguistic studies, integrating new disciplines into school curricula such as spelling and dictation. According to exhaustive studies that we have done in the field of spellchecking errors made in typing Arabic texts, very few research works that deal with typographical errors specifically caused by the insertion or missing of the blank-space in words. On the other hand, spelling correction software remains ineffective for handling this type of errors. Failure to process errors due to the insertion/missing of blank-space between and in words leads and brings us back to situations of ambiguity and incomprehension of the meaning of the typed text. To remedy this limitation of correction, we propose in this article an ad-hoc probabilistic method which is based jointly on two approaches. The first approach treats the errors due to deletion or missing of blank-space between or inside words, while the second puts emphasis in correcting space insertion errors in a word of course in addition to other kinds of elementary editing errors (addition, deletion, permutation of characters). Our new approach combines edit distance with n-gram language models to correct the errors already mentioned. Our new approach gave an accuracy rate that reaches 98,14% for missing blank-space errors (noted MBSE) and 89,5% for insertion blank-dspace errors (noted IBSE), which gives an average correction rate of around 95,26%. These results are very encouraging and show the interest and the importance of our approach.

*This is an open access article under the [CC BY-SA](#) license.*



## Corresponding Author:

Hicham Gueddah

Intelligent Processing and Security of Systems team- FSR, E.N.S, Mohammed V University

B.P: 8007, Avenue des Nations Unies-Agdal, Rabat, Morocco

h.gueddah@um5r.ac.ma

## 1. INTRODUCTION

Since the computer age, the new information and communication technologies have continued to advance and to evolve everyday. This revolution and progress is accompanied with an enormous amount of information that is daily generated and stored on media (electronic journals, emails, blogs, briefs, tutorials, speeches). Since then, a new style of reading and writing has emerged. In practice, the notion of spelling error should increasingly take a more privileged place in typing texts. Often times we type a text speeding up without effective control over what has been written. Currently, and thanks to spell checkers, omnipresent in all word processing editors, emails, information retrieval engines, applications of social media, the editor can

review his typed text, correct marked errors and therefore improve and assist in correct and healthy writing of spelling errors (typos) in order to resolve ambiguity in the typed text. In the area of natural language processing (NLP), the research axis of spelling correction remains the most important and oldest. The first research works date back to the 1960s [1]. Building automatic spell checking systems is one of the oldest applications of NLP technic, since, according to Mitton [2], the first automatic detection systems appeared at the beginning of the 1960. Finding solutions to the problem of spell-checking text has long been a challenge. Several researchers have investigated the problem and, through their efforts, various technics and many algorithms have emerged. Error detection involves finding spelling incorrect words in a given text. While the correction phase consists in proposing the solutions that are near to the erroneous word [3].

The first research in the area of spelling correction attempted to modelize the notion of spelling error. The founding article for this modeling was proposed by Damerau [4]. According to a statistical study carried out on spelling errors out of context, Damerau considered that an error is a single or multiple combination of elementary editing operations relating to the insertion, deletion, permutation and transposition of characters in a lexical word. Since then, and based on this modeling, several approaches and algorithms have been proposed for the correction of spelling errors. Metric method initiated by Levenshtein remains the most suitable [5]. It makes it possible to compare in two while calculating the number of elementary editing operations for insertion, deletion and permutation to transform the compared into that of the comparator. This metric is known as the edit distance, which remains despite the relevant distance technic practically applied in all spell-checking functions as extensive to use in the Bioinformatics field. Subsequently, a succession of approaches were proposed to enrich the correction methods that we can classify them into: i) correction approach based on lexical similarity measurement: this category includes, for example, the edit distance, Jaccard distance Jaccard [6], Jaro distance [7], [8], Stoilos *et al.* distance [9]; ii) probabilistic correction approach such as approaches based on probabilistic finite state automata Oflazer [10], the alpha-code method Pollock *et al.* [11], Dice index method [12], or the correction method by n-grams decomposition, or even research-based approaches in lexicographic trees [13]; iii) hybrid correction approach: concept that consist in joining metric based spelling correction and one, which uses probabilistic language models, and learning corpora. It is in this sense that we have proposed a series of works on spellchecking dedicated to the Arabic language;

Our objectives in this research work were to improve the scheduling rate and the precision rate in the edit distance based correction process [14]-[16], as well as to integrate the level of morphological analysis in the spelling correction phase [17], as well as taking into consideration the context in the correction process [18]. For a decade, our research team has consistently presented a series of relevant approaches in the field of spelling correction for errors made in typing Arabic texts [19], [20]. From our recently published study, we raised that all spellchecking systems remain ineffective in correcting some types of typing errors. We specifically cite here errors arising from improperly inserting and/or deleting the keyboard space character in a lexical word. Statistically speaking, several studies have shown that space errors occupy a considerable proportion among other types of errors, according to a statistical study on spelling errors made in typing Ordu texts, Naseem and Hussain [21] proved that 23% of these errors are of the blank-space insertion/deletion type, as long as Pedler [22] has identified that 8% of the errors made on an English language learning corpus, are of the space error type. Other than, these spellcheckers only provide solutions separately to segments marked as non-dictionary words. However, failure to process this type of error can lead to situations of ambiguity and incomprehension of the meaning of the typed text. This category of errors can be made in different situations, as an example: often the text is typed quickly without control; the writer can remove space between two successive words or insert space inside the word. Also following optical character recognition (OCR) application can cause such a situation deleting space between words. As we also raised placing/concealing space when converting PDF documents to another WinWord document.

Error classification: first, we can define an error as being any lexical form which does not correspond to any form stored in the dictionary or generated. Based on several statistical studies of learning corpora, several classifications of spelling errors have been proposed. The following classes may be mentioned in particular to: i) typing errors: mainly due to doubling characters, inverting characters, omitting characters, hitting a neighboring key; ii) editing errors: due to copy / paste operations, repetition of a word and/or a sentence, absence of a word, or even breaking of a word; iii) usual spelling errors (dyslexia): usually due to the lack of correspondence between oral and written, and committed on less frequent words in the dictionary; iv) grammatical errors: these are lexically correct words, but they do not respect the grammatical rules of a given language (rules of agreement and grammar ); v) segmentation errors: this type of error that interests us in this

study. They come from the omission of a space (fusion) or the insertion of a space segmentation) [22].

As pointed out in the introduction, errors due to the space character are caused by either deleting the space or inserting the space: i) missing blank-space errors: In this case, of error, the editor omits to insert space between two words in succession which subsequently results in a merger between two words considered as an entry not belonging to the lexicon. Converting documents from one software to another, such as one PDF document to another Word document and vice versa, this type of errors often times lead to word blank-space deletion throughout the converted document. Example: "naturallanguage" instead of "natural language"; ii) blank-space insertion errors: Often we type the text by accelerating without taking into account what has been typed, the writer can inadvertently insert one or more spaces inside a word, which causes a segmentation of the word in several sequences that can be lexical entries or erroneous sequences. We have pointed out that space insertion errors come from converting PDF file type to another Word document and vice versa, or even if we are trying to open a document written in WinWord with an earlier version. For example, instead of obtaining the word "technology" after conversion, a white space may be inserted and getting both sequences "techn" and "ology" that are incorrect lexical entries.

In reality, this kind of blank-space insertion errors disrupts the syntax and semantics of the whole sentence, especially when the segments generated are lexical words, which is the case for the Arabic language for example. In order to correct these errors, the editor's intervention is required to detect them first and then correct them. For example, the meaning of this sentence "Vaccination against COVID-19" is not the same after inserting the space in the word vaccination, "Vac ci nation against COVID-19". Based on a study we did on spelling errors due to insertion/missing blank-space, there are few studies that have looked at errors due to blank-space. Among these works are those which rely on generating all possible sequence of the erroneous word followed by checking if each partition exists in the dictionary [23], [24]. Alkanhal *et al.* [25] presented a method which merges the different neighboring words to the erroneous word. The result of this procedure is a list of the different possible combinations of this merge and subsequently select the correct merges from the wrong one.

## 2. METHOD

In this article, we propose a probabilistic metric method based on the edit distance and the n-gram language models to better correct errors caused by blank-space (insertion and deletion), in combination with other types of errors like elementary editing errors (addition, deletion and permutation). To deal with editing errors added to errors due to blank-space insertion and missing, we have developed a new approach that corrects this type of error. The latter combines the approach of correcting blank-space missing errors [26] with another that corrects blank-space insertion errors. In the following section, we will present these two approaches in detail.

### 2.1. Correction approach for missing blank-space errors

We recall here that our method of correcting blank-space missing errors is based on an adaptation of the edit distance. The principle is to detect where the space has been omitted and then proceed to the actual correction Yousfi *et al.* [26]. The position (noted  $pos.space$ ) where we will insert space in the erroneous word  $w_{err}$  is modeled by the rule we have proposed (1).

$$pos.space_i = \arg \min_{j=1, \dots, n} D_{ed}(w_{err}^j, w_i) \quad (1)$$

Where:

- $w_i$  is a word of the vocabulary  $V$  and  $D_{ed}$  is the edit distance.
- $w_{err}^j = e_1 e_2 \dots e_j$  is the part of  $w_{err}$  from the first character to the  $j^{th}$  character.
- $e_i$  is the  $i^{th}$  character of the  $w_{err}$
- $n$  is the word length of  $w_{err}$

After detecting where the space has been omitted, we subsequently obtain fragments:

- $w_{1pos.space_i} = e_1 e_2 \dots e_{pos.space_i}$  and  $w_{2pos.space_i} = e_{pos.space_i+1} \dots e_n$
- $w_{1pos.space_i} + w_{2pos.space_i} = w_{err}$

where:  $e_{pos.space_i+1}$  is the character of the word  $w_{err}$  which is at the position  $pos.space_i + 1$ .

In the second phase, we check if the two words are in the lexicon. If not, then we must correct the two resulting segments  $w_{1pos.space_i}$  and  $w_{2pos.space_i}$  which may also contain other basic editing errors. To correct at the same time the blank-space missing errors and other kinds of editing errors, we defined a new metric, noted  $D_{blsp}$ , based on the edit distance(2).

$$D_{blsp}(w_{err}, w_i) = Min[D_{ed}(w_{err}, w_i); \min_{w_j \in V} D_{ed}(w_j, w_{1space_i}) + \min_{w_j \in V} D_{ed}(w_j, w_{2space_i})] \quad (2)$$

The best solutions are those that satisfy (3):

$$\min_{w_i \in V} D_{blsp}(w_{err}, w_i) \quad (3)$$

## 2.2. Correction approach for insertion blank-space errors

To correct the errors caused by the blank-space insertion, we have developed a new method which relies on the edit distance and the n-gram language models [27]-[29]. This made it possible to define a new probabilistic measure to correct this kind of error. The principle being to detect that the marked error is due to an insertion of blank-space in a word and then to proceed to the effective correction. For the insertion error detection phase, we studied the different scenarios, such as: i) if two successive words  $w_1$  and  $w_2$  are wrong, then the probability is very high that the keyboard space character has been inserted in the word. In this situation, we join the two sequences and we proceed to correct  $w_1 + w_2$  as a single entity. Example of inserting a space in the word "reda biliti",  $w_1$ ="reda" and  $w_2$ ="biliti", we merge the two sequences and go to the correction of the merge "redabiliti"; ii) if we have a word consisting of only one character, then very likely that there is a wrong insertion of the space in the word "Irresistible" which causes for example "I rresistible". However, this is not always the case with for example after segmentation by the space character of the word "Iris" which gives two correct segments "I" and "ris"; iii) the last case, if there is any wrong word and the next word is correct, in this situation the space character may have been inserted in the word.

To identify these different scenarios, we present here two methods: i) process 1: the insertion of a blank-space error is only handled when there are two consecutive error words; ii) process 2: a erroneous word and regardless of the next word, adding the simple correction of this wrong word. We add all list solutions of the error caused by incorrect insertion of blank-space between this erroneous word and the two neighboring words.

## 2.3. Correcting blank-space insertion errors according to process 1

Let  $T = w_1 w_2 \dots w_n$  be a text composed of a set of arabic words as following, and suppose that we have two successive erroneous words  $w_i$  and  $w_{i+1}$  in this text. Our proposal consists in joining  $w_i$  and  $w_{i+1}$  ( $w_i + w_{i+1}$ ), and checking whether this ( $w_i + w_{i+1}$ ) exists in the lexicon of our system. If so, we keep this fusion as a potential solution, otherwise we use our measure  $D_{Wed}$ , which is a kind of weighting between the edit distance and the bi-gram language models, The distance  $D_{Wed}$  is defined as (4) and (5):

$$D_{Wed}(w_i, w_k) = Min[\frac{D_{ed}(w'_i, w_k)}{Pr(w_k/w_{i-1})} + \frac{D_{ed}(w_{i+1}, w_k)}{Pr(w_k/w_i)}, \frac{D_{ed}(w_i + w_{i+1}, w_k)}{Pr(w_k/w_{i-2})}] \quad (4)$$

The best solutions are those that check:

$$\min_{w_k \in Lexique} D_{Wed}(w_i, w_k) \quad (5)$$

## 2.4. Correcting blank-space insertion errors according to process 2

To correct the spelling errors, taking into account as well the errors due to blank-space for the case mentioned in process 2, we apply a  $D_{Wed}$  measure, which is defined this time as (6).

$$D_{Wed}(w_i, w_k) = Min[\frac{D_{ed}(w_i, w_k)}{Pr(w_k/w_{i-1})}, \frac{D_{ed}(w_{i-1} + w_i, w_k)}{Pr(w_k/w_{i-2})}, \frac{D_{ed}(w_i + w_{i+1}, w_k)}{Pr(w_k/w_{i-1})}] \quad (6)$$

$w_{i-1} + w_i$ : the both concatenated words  $w_{i-1}$  and  $w_i$

The best corrections of the erroneous word  $w_i$  are given by the following formula:

$$\arg \min_{w_k \in \text{Lexique}} D_{Wed}(w_i, w_k) = \arg \min \left[ \frac{D_{ed}(w_i, w_k)}{Pr(w_k/w_{i-1})}, \frac{D_{ed}(w_{i-1} + w_i, w_k)}{Pr(w_k/w_{i-2})}, \frac{D_{ed}(w_i + w_{i+1}, w_k)}{Pr(w_k/w_{i+1})} \right] \quad (7)$$

Example: Let this sentence containing a fragment word, "Python is an interpreted language programming". We have two successive erroneous words after insertion blank-space in the word "interpreted", the processing is therefore carried out by both methods.

i) Process 1:

a) The spelling correction of the two erroneous sequences "interp" and "reted" are:

- "interp" = {inter, inters, enter, instep, intrepid} and
- "reted" = {rated, rented, rested, reed, retied}. the minimum distance for both is 1, so the sum 1+1= 2.

b) The solution "interpreted", after deleting space between sequences "interp" and "reted" we obtain this word which is a lexicon entry.

c) Using the distance  $D_{Wed}$ , the minimum value of 0 and 2 is 0, and like that, the best solution is "interpreted" ("Python is an interpreted language programming").

ii) Process 2:

a) The first erroneous sequence is "interp", so the three sets without taking into account the n\_gram language models in the calculation:

b) Corrections of the erroneous word "interp" = {inter, inters, enter, instep, intrepid}

c) Corrections of the wrong word "aninterp" = {empty}, in reality no suggestion.

d) The suggestion word "interpreted", this word is a lexicon entry.

The last set contain the best solution of the blank-space insertion error.

### 3. RESULTS AND DISCUSSION

To test and show the effectiveness of our Ad-hoc correction method dedicated to the correction of errors due to the insertion/missing of blank-space in a word while taking into account at the same time the other types of elementary editing (addition, deletion and permutation), we built a corpus of 1000 paragraphs extracted from the Wikipedia site. For the space deletion errors in combination with the editing errors, we built a corpus of 3000 errors. For insertion errors and other types of editing errors, we randomly make errors in words inside the paragraphs of our corpus. The number of errors in this corpus is 1500 insertion errors. For the case of errors of deletion combined with editing errors, we proceeded to correct these errors using our approach described in this article, but we also took a very difficult initiation by comparing our approach to correcting with the spellchecker integrated into the WinWord editor.

This of course taught us a lot of time, but the results were surprising at the level of correction rate which reached 98.14% for our method against only 42.53% for WinWord. For the case of space insertion errors in combination with editing errors, we have proceeded to correct these errors using the approach described in section 4.2, always comparing them with the spellchecker built into WinWord. The results obtained showed that our probabilistic metric is the best compared to the WinWord spellchecker with a correction rate reaching 89.5% against only 19.12% for WinWord. From what we found that the patches, not only the one built into WinWord, only offers solutions to segments marked wrong after inserting the space in the word and ignore the rest of the words in the sentence.

### 4. CONCLUSION

In this contribution, we presented the details of our spelling correction system which is based on an ad-hoc method for the correction of errors due to the insertion/missing of the blank-space character in words by

taking into consideration other types of basic typing errors. This ad-hoc method relies on a probabilistic metric underlying the edit distance and probabilistic language models for correcting errors due to insertion/missing of the blank-space. This method of correction is very effective in removing ambiguity and misunderstanding of the text that can be caused by errors due to space. The experimental results we have obtained on hundreds of space errors in combination with other types of editing errors are very satisfactory and express the validity and efficiency of the design choices we have predicted at the start of our study. Another advantage of our correction method is the very high correction rate compared to another recognized commercial corrector.





## REFERENCES

- [1] K. Kukich, "Techniques for automatically correcting words in text," *ACM Computing Surveys (CSUR)*, vol. 24, no. 4, pp. 377-439, 1992, doi: 10.1145/146370.146380.
- [2] R. Mitton, "Fifty years of spellchecking," *Writing Systems Research*, vol. 2, no. 1, pp. 1-7, 2010, doi: 10.1093/wsr/wsq004.
- [3] C. Enguehard, K. Soumana, M. Mathieu, M. Issouf, and L. S. Mamadou, "Towards the computerization of some west-african languages(in French)," in *JEP-TALN-RECITAL 2012, Workshop TALAf 2012: Traitement Automatique des Langues Africaines (TALAf 2012: African Language Processing)*, 2011, pp. 27-40.
- [4] F. J. Damerau, "A Technique for computer detection and correction of spelling errors," *Communications of the ACM*, vol. 7, no. 3, pp. 171-176, 1964, doi: 10.1145/363958.363994.
- [5] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 8, no. 10, pp. 707-710, 1966.
- [6] P. Jaccard, "The distribution of the flora in the alpine zone," *New Phytologist*, vol. 11, pp. 37-50, 1912, doi: 10.1111/j.1469-8137.1912.tb05611.x.
- [7] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the census of Tampa," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 414-420, 1985, doi: 10.2307/2289924.
- [8] W. E. Winkler, "The state of record linkage and current research problems," *Statistical Society of Canada, Proceedings of the Section on Survey Methods*, 1999, pp. 73-79.
- [9] G. Stoilos, G. Stamou, and S. Kollias, "A string Metric for Ontology Alignment," *Proc. 4th International Semantic Web Conference (ISWC)*, 2005, pp. 624-37, doi: 10.1007/11574620\_45.
- [10] K. Oflazer, "Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction," *Computational Linguistics*, vol. 22, no. 1, pp. 73-98, 1996.
- [11] J. J. Pollock and A. Zamora, "Automatic spelling correction in scientific and scholarly text," *Communications of the ACM*, vol. 27, no. 4, pp. 358-68, 1984, doi: 10.1145/358027.358048.
- [12] R. C. Angell, G. E. Freund, P. Willett, "Automatic spelling correction using a trigram similarity measure," *Information Processing and Management*, vol. 19, no. 4, pp. 255-261, 1983, doi: 10.1016/0306-4573(83)90022-5.
- [13] L. Karttunen and K. Oflazer, "Introduction to the special issue on finite state methods in NLP," *Computational Linguistics*, vol. 26, no. 1, pp. 1-2, 2000, doi: 10.1162/089120100561593.
- [14] H. Gueddah, A. Yousfi and M. Belkasmi, "Introduction of the weight edition errors in the Levenshtein distance," *International Journal of Advanced Research in Artificial Intelligence*, vol. 1, no. 5, pp. 30-32, 2012, doi: 10.14569/IJARAI.2012.010506.
- [15] H. Gueddah and A. Yousfi, "Impact de la proximité et de la similarité inter-caractère Arabe sur la correction orthographique," *Proceeding of the 8th International Conference on Intelligent Systems : Theories and Applications- SITA 13*, 2013, pp. 244-246.
- [16] M. Nejja and A. Yousfi, "A lightweight system for correction of Arabic derived words," in *Mediterranean Conference on Information & Communication Technologies*, 2015, pp. 131-138, doi: 10.1007/978-3-319-30301-7\_14.
- [17] M. Nejja and A. Yousfi, "Contexts impact on the automatic spelling correction," *International Journal of Artificial Intelligence and Soft Computing archive*, vol. 6, no. 1, pp. 56-74, 2017, doi: 10.1016/j.procs.2015.12.055.
- [18] H. Bakkali, H. Gueddah, A. Yousfi and M. Belkasmi, "For an independent SpellChecking system from the Arabic language vocabulary," *Proceeding of International Journal of Advanced Computer Science and Applications*, vol. 5, no. 1, pp. 114-116, 2014, doi: 10.14569/IJACSA.2014.050115.
- [19] C. B. O. Zribi and M. B. Ahmed, "Efficient automatic correction of misspelled arabic words based on contextual information", *Lecture Notes in Computer Science*, vol. 2773, pp. 770-777, 2003, doi: 10.1007/978-3-540-45224-9\_104.
- [20] M. Attia, P. Pecina, Y. Samih, K. Shaalan and J. V. Genabith, "Arabic spelling error detection and correction," *Natural Language Engineering*, vol. 22, no. 5, pp. 751-773, 2015, doi: 10.1017/S1351324915000030.
- [21] T. Naseem, S. Hussain, "A novel approach for ranking spelling error corrections in Urdu," *Language Resources and Evaluation*, vol. 41, no. 2, pp. 117-28, 2007, doi: 10.1007/s10579-007-9028-6.
- [22] J. Pedler, "Computer correction of real-word spelling errors in dyslexic text," Thèse de doctorat, University of London, 2007.
- [23] T. Naseem, S. Hussain, "A Hybrid approach for Urdu spell checking," MSc thesis, National University of Computer & Emerging Sciences, Pakistan, 2004.
- [24] M. A. Maha, J. T. William, "Automatic correction of Arabic dyslexic text," *Computers 2019*, vol. 8, no. 1, p. 19, 2019, doi: 10.3390/computers8010019.
- [25] M. I. Alkanhal, M. A. Al-Badrashiny, M. M. Alghamdi and A. O. Al-Qabbany, "Automatic stochastic Arabic spelling correction with emphasis on space insertions and deletions," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 2111-2122, 2012, doi: 10.1109/TASL.2012.2197612.
- [26] A. Yousfi, L. Auragh, H. Gueddah, M. Nejja, "Spelling correction for the Arabic language -space deletion errors," in *International Workshop on Artificial Intelligence for Natural Language Processing, Procedia Computer Science*, 2020, pp. 568-574, doi: 10.1016/j.procs.2020.10.080.





- [27] G. Dalkılıç and Y. Çebi, "Turkish spelling error detection and correction by using word n-grams," *5<sup>th</sup> International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control*, 2009, pp. 1-4, doi: 10.1109/ICSCCW.2009.5379481.
- [28] M. M. Al-Jefri and S. A. Mahmoud, "Context-sensitive Arabic spell checker using context words and n-gram language models," *Taibah University International Conference on Advances in Information Technology for the Holy Quran and Its Sciences*, 2013, pp. 258-263, doi: 10.1109/NOORIC.2013.59.
- [29] S.L. Aouragh, A. Yousfi, S. aaroussi, H. Gueddah, and M. Nejja, "A New Estimate of the n-gram Language Model," *Procedia Computer Science*, vol. 189, pp. 211-215, 2021, doi: 10.1016/j.procs.2021.05.111.

## BIOGRAPHIES OF AUTHORS







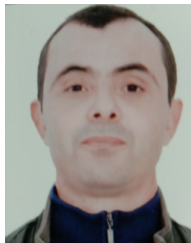
**Hicham Gueddah**     is an Assistant Professor in computer science at: Department of Computer Science-ENS, Mohammed V University in Rabat- Morocco, joining the Intelligent Processing and Systems Security team- FSR, Mohammed V University since 2019. His research area and topic is Natural Language Processing (NLP), Text Mining and Deep Learning, especially the research axis of automatic correction. He has published several works in automatic correction. Further info on his homepage: <https://www.researchgate.net/profile/Hicham-Gueddah>. He can be contacted at email: [h.gueddah@um5r.ac.ma](mailto:h.gueddah@um5r.ac.ma)







**Mohamed Nejja**     received his Ph.D. in Computer Science and Engineering from the National High School of Computer Science and Systems Analysis (ENSIAS) at Mohamed V University in Rabat, Morocco, in 2019. His areas of research interests include Natural Language Processing, Machine Learning. He can be contacted at email: [mohammed.nejja@gmail.com](mailto:mohammed.nejja@gmail.com)







**Said Iazzi**     is a doctoral at the Mohammed V University of Rabat, Discipline Engineering Sciences. His researches are in fields of computer sciences, Automatic Arabic Language Processing, IA, Data Science and Machine Learning. He can be contacted at email: [iazzisaid@yahoo.fr](mailto:iazzisaid@yahoo.fr)



**Abdellah Yousfi**     is Professor at the Faculty of Law, Economics and Social Sciences Souissi at Mohamed V University in Rabat, since 2007. He is member of the Information, Communication, Embedded Systems and Natural language processing Team at the National High School of Computer Science and Systems Analysis (ENSIAS) Mohamed V University in Rabat, Morocco. His research interests include creation of corpora for the Arabic language, Arabic speech recognition, Arabic handwriting recognition and correction of Arabic spelling errors. He can be contacted at email: [a.yousfi@um5r.ac.ma](mailto:a.yousfi@um5r.ac.ma)



**Si Lhoussain Aouragh**     is permanent qualified professor at the National High School of Computer Science and Systems Analysis (ENSIAS) at Mohamed V University in Rabat, since 2021. From 2016 to 2021 Accredited Professor at the Faculty of Legal, Economic and Social Sciences, Salé. President of the Association of Arabic Language Engineering in Morocco. He can be contacted at email: [l.aouragh@um5r.ac.ma](mailto:l.aouragh@um5r.ac.ma)